XLIII CILAMCE
21-25, November, 2022
FOZ DO IGUAÇU - PR - BRAZIL

Virtual Element Method for 3D Poisson Equation

Tiago F. Moherdaui[1], Alfredo G. Neto[1]

[1]*Dept. of Structural and Geotechnical Engineering, University of São Paulo*
Av. Prof. Almeida Prado, 83, 05508-070, São Paulo, Brazil
tiago.moherdaui@usp.br, alfredo.gay@usp.br

Abstract. This work reports some of the challenges met with when implementing the Virtual Element Method (VEM) for Poisson's problem in a three-dimensional setting. The method's generalization of the Finite Element Method (FEM) for polytopal shapes and arbitrary order introduces many challenges, such as integration over the element, mesh generation, etc. There is a significant increase in complexity when going from a two-dimensional to a three-dimensional implementation of the method, thus the main topic of this work. The Poisson problem is chosen because of its simplicity, allowing the focus to remain on the method itself. The main comparisons are between the two and three-dimensional versions of the method, as well as between the latter and the equivalent Finite Element Method.

Keywords: Virtual element method, Finite element method

1 Introduction

The Virtual Element Method (VEM) was first introduced in Beirão da Veiga, et al. [1], and consists in a generalization of the Finite Element Method (FEM) for polytope elements of arbitrary order in a unified formulation. Along with the method's basic principles, a paper laying down the theoretical groundwork for linear elasticity was also published (Beirão da Veiga, Brezzi and Marini [2]). Ever since, the method has been further developed for linear and nonlinear elastostatic problems, both for 2D (e.g., Artioli, et al. [3], Mengolini, Benedetto and Aragón [4], and Reddy and van Huyssteen [5]) and 3D (e.g., Chi, Beirão da Veiga and Paulino [6]), for elastodynamics (e.g., Park, Chi and Paulino [7], and Cihan, et al. [8]), and in combination with Discrete Element Methods (e.g., Gay Neto, et al. [9]), to mention a few applications.

The method's presentation in Beirão da Veiga, et al. [1], as well as the more implementation-oriented version in Beirão da Veiga, et al. [10], is done in the context of the Poisson problem, which provides a simpler framework for an introduction. The two-dimensional version of the problem has already been explored by the authors in the context of St. Venant's torsion, see Moherdaui and Gay Neto [11]. The present paper can be taken as a followup with the application of the method for the three-dimensional version of the problem. The additional dimension represents an increase in complexity for any method, even more so for the VEM due the method's geometrical versatility.

The goal of this paper is to give an overview of these difficulties, as well as to compare them with their analog counterparts in the context of the Finite Element Method.

The rest of the paper is organized as follows. Section 2 poses Poisson's problem in two and three dimensions, followed by a brief introduction to the VEM and its main components in section 3. In sequence, section 4 presents an examination of some particular characteristics of the method and the challenges found when dealing with the additional dimension. Finally, section 5 presents some final thoughts and considerations on the topic.

2 Poisson's Equation

Poisson's equations define their solution by its Laplacian, value and/or flux along surfaces. Among its applications are mostly problems that deal with potential fields (e.g., electrostatics, newtonian gravitation) but also other problems such as St. Venant's warping (Moherdaui and Gay Neto [11]), and the shape functions for the virtual element method (Moherdaui and Gay Neto [12]).

The problem is posed in its differential formulation as

$$\begin{cases} -\Delta u(\mathbf{x}) = f(\mathbf{x}), & \mathbf{x} \in \Omega \\ \nabla u(\mathbf{x}) \cdot \mathbf{n} = g(\mathbf{x}), & \mathbf{x} \in \partial\Omega_n \\ u(\mathbf{x}) = \bar{u}(\mathbf{x}), & \mathbf{x} \in \partial\Omega_d \end{cases} ,$$

where $u : \Omega \to \mathbb{R}$ is the solution, $f : \Omega \to \mathbb{R}$ is the opposite of the prescribed value for the solutions laplacian, $g : \partial\Omega_n \to \mathbb{R}$ is the prescribed Neumann boundary condition and $\bar{u} : \partial\Omega_d \to \mathbb{R}$ is the prescribed Dirichlet boundary condition.

The corresponding weak formulation is

$$\begin{cases} \int_\Omega \nabla u \cdot \nabla \delta u d\mathbf{x} = \int_\Omega f \delta u d\mathbf{x} + \int_{\partial\Omega_n} g \delta u d\sigma \\ u(\mathbf{x}) = \bar{u}(\mathbf{x}), \quad \mathbf{x} \in \partial\Omega_d \end{cases} , \tag{1}$$

where $\delta u : \Omega \to \mathbb{R}$ is an arbitrary test function.

Both formulations being valid for both $\Omega \in \mathbb{R}^2$ and $\mathbb{R}^3$.

3 Virtual Element Method

This section introduces the Virtual Element Method in the context of Poisson's equations for two and three dimensions. The notation is a combination of Beirão da Veiga, et al. [10] and Chi, Beirão da Veiga and Paulino [13].

Whereas classic FEM imbues each element E with a polynomial local approximation space, the essence of the VEM consists in providing a procedure to construct a non-polynomial local approximation space $V_k(E)$ based on the element's shape and associated order k. The approximation space contains the full polynomial space of the element's associated order $V_k(E) \supseteq P_k(E)$.

The degrees of freedom are chosen to allow the construction of a projection operator $\Pi_E^\nabla : V_k \to P_k$ onto the polynomial subspace without requiring too much knowledge of these functions.

$$\begin{cases} \int_E \nabla(\Pi_E^\nabla v) \cdot \nabla p_k d\mathbf{x} = \int_E \nabla v \cdot \nabla p_k d\mathbf{x} \;\; \forall p_k \in P_k(E) & \\ \sum_{\mathbf{x}_v \in E} \Pi_E^\nabla v(\mathbf{x}_v) = \sum_{\mathbf{x}_v \in E} v(\mathbf{x}_v) & \text{if } k = 1 \\ \int_E \Pi_E^\nabla v d\mathbf{x} = \int_E v d\mathbf{x} & \text{if } k \geq 2 \end{cases} . \tag{2}$$

The first term is the projection (Π^∇) based on the weak formulation of the problem eq.(1), whereas the other two are a supplemental projection ($P0$), meant to circumvent the kernel of the first one and provide a constant term for the projection. The projection is used to compute the stiffness matrix and load term, the former of which is split into a consistency term exactly computed and a stabilization term which is approximated.

Remark 1: Domain notation. This section presents the method for two and three dimensions. When referring specifically to a three-dimensional domain, it will be explicitly denoted by P. Whereas two-dimensional domains, either a 2D VEM domain or a face of a 3D VEM domain, will be denoted by F. When a piece of formulation is valid for both cases, the notation E is used to denote the domain for both the 2D and 3D formulations.

3.1 2D VEM

To introduce the space for a general polygonal element F, first a preliminary space $\tilde{V}_k(F)$ must be presented

$$\tilde{V}_k(F) \doteq \{v \in H^1(F) : v|_e \in P_k(e) \forall e \in \partial F, \; \Delta v \in P_k(F)\}. \tag{3}$$

This space contains functions whose Laplacian (Δv) and trace ($v|_e \; e \in \partial F$) are both kth order polynomials. The associated degrees of freedom are

- the values of v at vertices: $v(\mathbf{x}_v), \; \forall \mathbf{x}_v \in F$; (4)
- the values of v at the edge points: $v(\mathbf{x}_e^l), \; \forall e \in \partial F, \; l = 1, \ldots, k-1$; (5)

- the moments of v up to order $k-2$: $\int_F vp_{k-2}dF;\ \forall p_{k-2} \in P_{k-2}(F).$ (6)

The chosen edge points are those, aside from the vertices, that make up the Gauss-Lobatto quadrature rule of order $2k-1$.

Integration by parts applied to the right hand side of the first equation in eq. (2) leads to

$$\int_F \nabla v \cdot \nabla p_k d\mathbf{x} = -\int_F v\Delta p_k d\mathbf{x} + \int_{\partial F} v(\nabla p_k \cdot \mathbf{n})d\sigma, \tag{7}$$

which can be exactly computed using eq. (6) for the first term and Gauss-Lobatto integration, with eqs. (4) and (5), for the second.

The local approximation space $V_k(F)$ is defined as

$$V_k(F) \doteq \left\{ v \in \tilde{V}_k(F) : \int_F vq d\mathbf{x} = \int_F (\Pi_F^{\nabla} v)q d\mathbf{x}\ \ \forall q \in (P_k/P_{k-2}(F)) \right\}, \tag{8}$$

where $(P_k/P_{k-2}(F))$ stands for the polynomial space in $P_k(F)$ that is L^2 orthogonal to P_{k-2}. The same set of degrees of freedom applies to this space.

3.2 3D VEM

For the three-dimensional counterpart on a general polyhedral element P, the preliminary space $\tilde{V}_k(P)$ is

$$\tilde{V}_k(P) \doteq \{v \in H^1(P) : v|_F \in V_k(F)\ \forall F \in \partial P,\ \Delta v \in P_k(P)\}. \tag{9}$$

The space is very similar to the one for 2D VEM, and the degrees of freedom are also a natural extension of those presented before

- values of v at vertices: $v(\mathbf{x}_v),\ \forall \mathbf{x}_v \in P$; (10)
- values of v at $v(\mathbf{x}_e^l),\ \forall e \in \partial F,\ l = 1, \ldots, k-1$; (11)
- the moments of v up to order $k-2$ on face F: $\int_F vp_{k-2}ds\ \ \forall p_{k-2} \in P_{k-2}(F)$ and $\forall F \in \partial P$; (12)
- the moments of v up to order $k-2$ in P: $\int_P vp_{k-2}d\mathbf{x}\ \ \forall p_{k-2} \in P_{k-2}(P).$ (13)

Hence, the approximation space $V_k(P)$ is defined as:

$$V_k(P) \doteq \left\{ v \in \tilde{V}_k(P) : \int_P vq d\mathbf{x} = \int_P (\Pi_P^{\nabla} v)q d\mathbf{x}\ \ \forall q \in (P_k/P_{k-2}(P)) \right\}. \tag{14}$$

In this case, the integration by parts of the right hand side becomes

$$\int_P \nabla v \cdot \nabla p_k d\mathbf{x} = -\int_P v\Delta p_k d\mathbf{x} + \int_{\partial P} v(\nabla p_k \cdot \mathbf{n})d\sigma, \tag{15}$$

whereas in this case the term in the volume remains exactly computable with eq. (13). The term on the boundary now explicits the importance of using V_k instead of $\tilde{V}_k$. The product $\nabla p_k \cdot \mathbf{n}$ is a polynomial of space P_{k-1}, which can be separated into a term in P_{k-2} and another in (P_k/P_{k-2}). The integral with the first term can be computed exactly with the use of the face degrees of freedom in eq. (12). However, the other term requires the additional property differentiating $V_k(P)$ and $\tilde{V}_k(P)$.

3.3 Important matrices

In the formulation presented in Beirão da Veiga, et al. [10], the fundamental quantities of the method (projector, stiffness matrix, load vector) are expressed in terms of three basic matrices: $\mathbf{B}$, $\mathbf{G}$ and $\mathbf{D}$.

$$B_{\alpha i} = \begin{cases} P0(\phi_i) & \alpha = 1 \\ \int_E \nabla m_\alpha \cdot \nabla \phi_i d\mathbf{x} & \alpha \geq 2 \end{cases} \tag{16}$$

$$G_{\alpha\beta} = \begin{cases} P0(m_\beta) & \alpha = 1 \\ \int_E \nabla m_\alpha \cdot \nabla m_\beta d\mathbf{x} & \alpha \geq 2 \end{cases} \tag{17}$$

$$D_{i\alpha} = dof_i(m_\alpha) \tag{18}$$

The matrices $\mathbf{B}$ and $\mathbf{G}$ define the matrix form of the projector which gives the coordinates of the resulting polynomial in term of the polynomial basis $\mathbf{\Pi}_*^\nabla$, i.e., $\Pi^\nabla \phi_i = \sum_\alpha (\Pi_*^\nabla)_{i\alpha} m_\alpha$

$$\mathbf{\Pi}_*^\nabla = \mathbf{G}^{-1}\mathbf{B}, \tag{19}$$

whereas for the form that gives the coordinates with respect to the degrees of freedom of the element ($\mathbf{\Pi}^\nabla = \mathbf{D}\mathbf{\Pi}_*^\nabla$), it just takes a change of basis matrix ($\mathbf{D}$).

Finally, the stiffness matrix $\mathbf{K}_e$ is split into consistency $\mathbf{K}_c$ and stability $\mathbf{K}_s$ terms

$$\begin{aligned} \mathbf{K}_e &= (\mathbf{\Pi}_*^\nabla)^T \tilde{\mathbf{G}} \mathbf{\Pi}_*^\nabla \\ \mathbf{K}_s &= (\mathbf{I} - \mathbf{\Pi}^\nabla)^T (\mathbf{I} - \mathbf{\Pi}^\nabla) \ \text{ if } \Omega \subset \mathbb{R}^2, \\ \mathbf{K}_s &= h_P (\mathbf{I} - \mathbf{\Pi}^\nabla)^T (\mathbf{I} - \mathbf{\Pi}^\nabla) \ \text{ if } \Omega \subset \mathbb{R}^3 \end{aligned} \tag{20}$$

where $\tilde{\mathbf{G}}$ is the same as $\mathbf{G}$ but with zeros in the first row, and h_P is the diameter of the polyhedron.

This concludes the presentation of the method, the load vector is purposefully left out, but can be found in the aforementioned works.

4 Analysis and comparison

This section comments on a few selected aspects of the implementation of the Virtual Element Method on a three-dimensional setting, whilst comparing with the respective aspects of 2D VEM and the FEM in general.

4.1 Geometry

The major feature of the VEM is the geometrical versatility of its elements. Even when considering only polyhedra with plane faces, the additional dimension leads to non-trivial complications.

One initial difference is the computational representation of a generic polyhedron. In the FEM, the fixed-shape elements allow their geometry to be defined by one sequence of a known number of vertices, ordered according to a predetermined convention. For 2D VEM, a general polygon may have any number of vertices, but it can be well defined by providing this information along with a list of the vertices in a pre-established order (e.g., counterclockwise). The general polyhedron, however, requires both an unordered list of vertices and the connectivity of the faces. This last information is usually a list of faces, where each face is a list of a variable size of vertices.

One non-trivial complication, then, comes when considering the geometry's triangulation. The decomposition of the domain into simplices is useful for tasks such as numerical integration (as simplices usually have well established quadrature rules) and visualization (as simplices tend to be rendering primitives for post-processing software).

In the case of the fixed-geometry FEM, the matter is trivial, the partition consists in invariant connectivity information which can be hardcoded. When considering 2D VEM's general polygon, there is a proven minimum number of triangles to partition a n_v-sided polygon: $n_v - 2$. And this minimal partition is easily obtained by algorithms such as the *Ear clipping algorithm*, originally introduced in Meisters [14], along with the previously mentioned minimum.

Whereas for general polyhedra, not only is there no proven minimum number of tetrahedra for the general case, there are cases of polyhedra which are not partitionable without the insertion of new vertices (e.g., Schönhardt polyhedron). There are algorithms that may produce triangulations for a given polyhedron without extra vertices, such as the *Greedy Peeling* and *Cone Triangulation* mentioned in Stojanovic [15]. Both of these work by taking out valid tetrahedra from the original polyhedron, which may leave a non-partitionable polyhedron as remainder, leading to a failed decomposition. A robust approach would require a first step of splitting the polyhedron into convex subpolyhedra, followed by their individual partitioning and a merger of those separate partitions. This is clearly a much more complicated process than the bi-dimensional equivalent, and there is always an uncertainty regarding the partitionability of the polyhedron.

4.2 Mesh generation

Finite element analysis software (e.g., ABAQUS, ADINA, ANSYS) usually include their own tools for mesh generation. There are also free software available that provide such services (e.g., GMSH - Geuzaine and Remacle [16]). The simplicity in computationally representing fixed-geometry elements is translated into the conventions of writing this information into mesh files.

The novelty of the VEM, as well as the previous Polygonal-FEM's, make the availability of polygonal-mesh generation software scarce when compared to the more well established Finite Element Method. A common method to generate polytopal meshes is to produce in-house algorithms to take a triangulation and generate an associated Voronoi Tessellation, with some additional work to preserve the boundary (approach used in Moherdaui and Gay Neto [11]). Another alternative is to use software that does this work for other purposes, such as Neper - Quey, Dawson and Barbe [17], a software for polycrystal generation and meshing. Nevertheless, there are some software and routines being developed to produce this type of tessellation with this end in mind (e.g., PolyMesher - Talischi, et al. [18])

Aside from the geometrical aspect of the mesh, there is another complication from the VEM in general that must be considered. In the FEM, every degree of freedom is associated with a position (a node). The same is only true of VEM for linear elements. For higher order elements in 2D, there are the internal degrees of freedom eq. (6), which are associated with the element, not with any position. In the case of 3D elements, there are internal degrees of freedom that are face-related eq. (12), and therefore shared between two elements, and those that are related only to one element eq. (13). There is no established convention for the representation for these degrees of freedom when communicating mesh-related information.

4.3 Monomials representation

Another characteristic of the method, aside from geometrical versatility, is its generality with respect to the element order. Whereas each finite element is usually tailored with a specific order in mind, the virtual elements are defined for the general order k. The degrees of freedom for both 2D and 3D VEM are defined based on the order of the element (e.g., edge degrees of freedom take the positions associated with Gauss-Lobatto quadrature, and inner degrees of freedom are associated with the space P_{k-2}). A complete implementation of the VEM should be malleable in this respect too.

In Beirão da Veiga, et al. [10], an indexing notation is presented for 2D monomials. For two dimensions, this correspondence is intuitive and can be visualized in a manner similar to Pascal's triangle. From this indexing notation, routines can be established to provide helpful information: e.g., exponent for each variable, order, which monomials are associated with its derivatives, which monomial is the outcome of the product of two others, etc. For the 3D case, this becomes a little less intuitive, but still achievable.

4.4 Integration

Integration plays a major role in both FEM and VEM, its efficacy ensures the convergence properties and its computational efficiency is important to avoid the stiffness matrix assembly process from becoming the bottleneck of the solution procedure.

Once more, the rigid geometry of finite elements allowed the development of specific quadrature procedures for each element, which have been tried and tested. For the VEM, although the shape functions themselves are not polynomials, the integration of polynomials is an essential part of the method, as can be seen with the matrices $\mathbf{B}$ and $\mathbf{G}$, eqs. (16) and (17).

A few approaches come to mind when thinking of numerical integration of arbitrary polynomials over general polytope domains. One is to triangulate the domain and accumulate the integrals over the simplices, for which quadrature rules are usually known and efficient. This is a suitable approach for 2D VEM, where the minimal triangulation is easily achievable, but for 3D VEM this might not be the most advisable method. Another is to apply Stoke's theorem, reducing the integral over the domain to one over the boundary, usually at the cost of increasing the polynomial order, which can become costly.

Chin and Sukumar [19] proposed a Homogeneous Numerical Integration method, which combines the latter approach with Euler's homogeneous function theorem, allowing the computation of integrals of homogeneous polynomials (the case of monomials) with successive applications of Stoke's theorem without increasing the degree of the integrand, requiring knowledge of the function and its derivatives on the boundary. This approach is suitable for both versions of VEM, especially when constructing a matrix like $\mathbf{G}$ where the integrals are computed for in increasing order of monomials.

4.5 Visualization

The visualization of information over a three-dimensional domain is always a challenge, regardless of method used to generate the information. Our lack of access to a fourth dimension renders us unable to visualize the full interior of a three-dimensional domain with information encoded as color, as well as to use the additional dimension as a tool to portray this information. Thus, other artifices must be employed such as the use of carefully positioned cut planes or isosurfaces and transparency.

For the Virtual Element Method, there is an additional complication: the solution is a combination of shape functions which are not polynomials nor known. A common workaround is the use of the polynomial projection of the solution, which may not only cause the visualization to lose global continuity, but may also hide more nuanced information in the solution. Although the projection is sufficient to convey the method the same convergence properties of the Finite Element Method, they do not accurately represent their corresponding shape functions when it comes to visualization, as can be seen in Moherdaui and Gay Neto [12]. This is still an open problem.

Additionally, the rendering results in polyhedra requires their triangulation. As, although there are visualization software which handle polygons, the authors have not found any that contain a polyhedral rendering primitive. Thus, the rendering must be done with tetrahedra as primitive, and the triangulation remains necessary.

5 Conclusions

The Virtual Element Method is a generalization of the Finite Element Method which handles both a general polytopal shape and arbitrary associated order k in one formulation. The shape/order versatility, however, is a double-edged blade that accompanies its own challenges when it comes to implementation. And these challenges become more difficult when the third dimension is added, as was expected.

To compare the availability of solutions and tools for the Virtual Element Method with the Finite Element Method is unfair. The former has been in existence for about eight decades, whereas the latter is barely completing its first one. Nevertheless, they are being developed. As the method has seen in the last few years an increasing number of publications and applications, it is only natural that more tools will be created (or extended) to aid the method. In the same way, as there are more people working on the same problems, it is expected that more solutions will come.

Acknowledgements. The authors acknowledge Vale S.A. for the support through Wheel-Rail Chair project. The second author acknowledges CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) under the Grant 304321/2021-4.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

References

[1] L. Beirão da Veiga, F. Brezzi, A. Cangiani, G. Manzini, L. D. Marini, and A. Russo. Basic Principles of Virtual Element Methods. *Mathematical Models and Methods in Applied Sciences*, vol. 23, n. 01, pp. 199–214, 2013a.
[2] L. Beirão da Veiga, F. Brezzi, and L. D. Marini. Virtual Elements for Linear Elasticity Problems. *SIAM Journal on Numerical Analysis*, vol. 51, n. 2, pp. 794–812, 2013b.
[3] E. Artioli, L. Beirão da Veiga, C. Lovadina, and E. Sacco. Arbitrary order 2D virtual elements for polygonal meshes: part I, elastic problem. *Computational Mechanics*, vol. 60, n. 3, pp. 355–377, 2017.
[4] M. Mengolini, M. F. Benedetto, and A. M. Aragón. An engineering perspective to the virtual element method and its interplay with the standard finite element method. *Computer Methods in Applied Mechanics and Engineering*, vol. 350, pp. 995–1023, 2019.
[5] B. D. Reddy and van D. Huyssteen. A virtual element method for transversely isotropic elasticity. *Computational Mechanics*, vol. 64, n. 4, pp. 971–988, 2019.
[6] H. Chi, L. Beirão da Veiga, and G. H. Paulino. Some basic formulations of the virtual element method (VEM) for finite deformations. *Computer Methods in Applied Mechanics and Engineering*, vol. 318, pp. 148–192, 2017.
[7] K. Park, H. Chi, and G. H. Paulino. Numerical recipes for elastodynamic virtual element methods with explicit time integration. *International Journal for Numerical Methods in Engineering*, vol. 121, n. 1, pp. 1–31, 2020.
[8] M. Cihan, B. Hudobivnik, F. Aldakheel, and P. Wriggers. Virtual Element Formulation for Finite Strain Elastodynamics. *Computer Modeling in Engineering I& Sciences*, vol. 129, n. 3, pp. 1151–1180, 2021.

[9] A. Gay Neto, B. Hudobivnik, T. F. Moherdaui, and P. Wriggers. Flexible polyhedra modeled by the virtual element method in a discrete element context. *Computer Methods in Applied Mechanics and Engineering*, vol. 387, 2021.

[10] L. Beirão da Veiga, F. Brezzi, L. D. Marini, and A. Russo. The Hitchhiker's Guide to the Virtual Element Method. *Mathematical Models and Methods in Applied Sciences*, vol. 24, n. 08, pp. 1541–1573, 2014.

[11] T. F. Moherdaui and A. Gay Neto. Virtual Element Method vs Finite Element Method for Prandtl's Solution of St. Venant's Torsion Problem. In *Proceedings of the XL Ibero-Latin American Congress on Computational Methods in Engineering*, 2019.

[12] T. F. Moherdaui and A. G. Neto. Virtual Element Method : Who are the virtual functions? In *Proceedings of the Ibero-Latin-American Congress on Computational Methods in Engineering*, pp. 1–7, Foz do Iguaçú, 2020.

[13] H. Chi, L. Beirão da Veiga, and G. H. Paulino. A simple and effective gradient recovery scheme and a posteriori error estimator for the Virtual Element Method (VEM). *Computer Methods in Applied Mechanics and Engineering*, vol. 347, pp. 21–58, 2019.

[14] G. H. Meisters. Polygons Have Ears. *The American Mathematical Monthly*, vol. 82, n. 6, pp. 648, 1975.

[15] M. Stojanovic. Algorithms for triangulating polyhedra into a small number of tetrahedra. *Matematicki Vesnik*, vol. 57, n. 1-2, pp. 1–9, 2005.

[16] C. Geuzaine and J.-F. Remacle. A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *Int. J. Numer. Meth. Engng.*, vol. 79, n. 11, pp. 1309–1331, 2017.

[17] R. Quey, P. Dawson, and F. Barbe. Large-scale 3D random polycrystals for the finite element method: Generation, meshing and remeshing. *Computer Methods in Applied Mechanics and Engineering*, vol. 200, n. 17-20, pp. 1729–1745, 2011.

[18] C. Talischi, G. H. Paulino, A. Pereira, and I. F. M. Menezes. PolyMesher: a general-purpose mesh generator for polygonal elements written in Matlab. *Structural and Multidisciplinary Optimization*, vol. 45, n. 3, pp. 309–328, 2012.

[19] E. B. Chin and N. Sukumar. An efficient method to integrate polynomials over polytopes and curved solids. *Computer Aided Geometric Design*, vol. 82, pp. 101914, 2020.